CS477 Formal Software Development Methods

Elsa L Gunter
2112 SC, UIUC
egunter@illinois.edu
http://courses.engr.illinois.edu/cs477

Slides based in part on previous lectures by Mahesh Vishwanathan, and by Gul Agha

March 27, 2013

Labeled Transition System (LTS)

A labeled tranistion system (LTS) is a 4-tuple (Q, Σ, δ, I) where

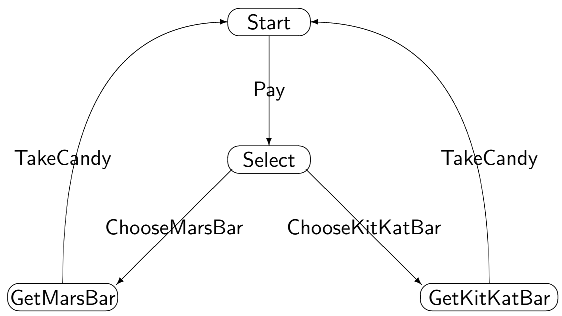
- Q set of states
  - Q finite or countably infinite
- Σ set of labels (aka actions)
  - Σ finite or countably infinite
- $\delta \subseteq Q \times \Sigma \times Q$ transition relation
- $I \subseteq Q$ initial states

Note: Write $q \xrightarrow{\alpha} q'$ for $(q, \alpha, q') \in \delta$.

Example: Candy Machine

- $Q = \{\text{Start}, \text{Select}, \text{GetMarsBar}, \text{GetKitKatBar}\}$
- $I = \{\text{Start}\}$
- $\Sigma = \{\text{Pay}, \text{ChooseMarsBar}, \text{ChooseKitKatBar}, \text{TakeCandy}\}$
- $\delta = \left\{ \begin{array}{l} (\text{Start}, \text{Pay}, \text{Select}) \\ (\text{Select}, \text{ChooseMarsBar}, \text{GetMarsBar}) \\ (\text{Select}, \text{ChooseKitKatBar}, \text{GetKitKatBar}) \\ (\text{GetMarsBar}, \text{TakeCandy}, \text{Start}) \\ (\text{GetKitKatBar}, \text{TakeCandy}, \text{Start}) \end{array} \right\}$

Example: Candy Machine



Predecessors, Successors and Determinism

Let (Q, Σ, δ, I) be a labeled transition system.

$$In(q, \alpha) = \{q' | q' \xrightarrow{\alpha} q\} \qquad In(q) = \bigcup_{\alpha \in \Sigma} In(q, \alpha)$$
$$Out(q, \alpha) = \{q' | q \xrightarrow{\alpha} q'\} \qquad Out(q) = \bigcup_{\alpha \in \Sigma} Out(q, \alpha)$$

A labeled tranistion system (Q, Σ, δ, I) is deterministic if

$$|I| \leq 1 \text{ and } |Out(q, \alpha)| \leq 1$$

Labeled Transition Systems vs Finite State Automata

- LTS have no accepting states
  - Every FSA an LTS - just forget the accepting states
- Set of states and actions may be countably infinite
- May have infinite branching

Exections, Traces, and Runs

- A partial exection is a finite or infinite alternating sequence of states and actions $\rho = q_0\alpha_1 q_1 \ldots \alpha_n q_n \ldots$ such that
  - $q_0 \in I$
  - $q_{i-1} \xrightarrow{\alpha_i} q_i$ for all i with q_i in sequence
- An exection is a maxial partial exection
- A finite or infinite sequence of actions $\alpha_1 \ldots \alpha_n \ldots$ is a trace if there exist states $q_0 \ldots q_n \ldots$ such that the sequence $q_0\alpha_1 q_1 \ldots \alpha_n q_n \ldots$ is a partial execution.
  - Let $\rho = q_0\alpha_1 q_1 \ldots \alpha_n q_n \ldots$ be a partial execution. Then $trace(\rho) = \alpha_1 \ldots \alpha_n \ldots$.

  A finite or inifnite sequence of states $q_0 \ldots q_n \ldots$ is a run if there exist actions $\alpha_1 \ldots \alpha_n \ldots$ such that the sequence $q_0\alpha_1 q_1 \ldots \alpha_n q_n \ldots$ is a partial execution.
  - Let $\rho = q_0\alpha_1 q_1 \ldots \alpha_n q_n \ldots$ be a partial execution. Then $run(\rho) = q_0 \ldots q_n \ldots$.

Example: Candy Machine

- Partial execution: $\rho = Start \cdot Pay \cdot Select \cdot ChooseMarsBar \cdot GetMarsBar \cdot TakeCandy \cdot Start$
- Trace: $trace(\rho) = Pay \cdot ChooseMarsBar \cdot TakeCandy$
- Run: $run(\rho) = Start \cdot Select \cdot GetMarsBar \cdot Start$

Program Transition System

A Program Transition System is a triple $(\mathcal{S}, T, init)$

- $\mathcal{S} = (\mathcal{G}, \mathcal{D}, \mathcal{F}, \phi, \mathcal{R}, \rho)$ is a first-order structure over signature $\mathcal{G} = (V, F, af, R, ar)$, used to interpret expressions and conditionals
- T is a finite set of conditional transitions of the form

  $$g \rightarrow (v_1, \ldots, v_n) := (e_1, \ldots, e_n)$$

  where $v_i \in V$ distinct, and e_i term in $\mathcal{G}$, for $i = 1 \ldots n$
- $init$ initial condition asserted to be true at start of program

Example: Traffic Light

$V = \{Turn, NSColor, EWColor\}$, $F = \{NS, EW, Red, Yellow, Green\}$ (all arity 0), $R = \{=\}$

$$\begin{array}{rrcl} T = & Turn = NS \wedge NSColor = Red & \rightarrow & NSColor := Green \\ & NSColor = Green & \rightarrow & NSColor := Yellow \\ & NSColor = Yellow & \rightarrow & (Turn, NSColor) := (EW, Red) \\ & Turn = EW \wedge EWColor = Red & \rightarrow & EWColor := Green \\ & EWColor = Green & \rightarrow & EWColor := Yellow \\ & EWColor = Yellow & \rightarrow & (Turn, EWColor) := (NS, Red) \end{array}$$

$init = (NSColor = Red \wedge EWColor = Red \wedge (Turn = NS \vee Turn = EW)$

Mutual Exclusion (Attempt)

$$\begin{array}{lll} P1 :: & m1 : & while\ true\ do \\ & m2 : & p11(*not\ in\ crit\ sect*) \\ & m3 : & c1 := 0 \\ & m4 : & wait(c2 = 1) \\ & m5 : & r1(*in\ crit\ sect*) \\ & m6 : & c1 := 1 \\ & m7 : & od \end{array} \qquad \begin{array}{lll} P2 :: & n1 : & while\ true\ do \\ & n2 : & p21(*not\ in\ crit\ sect*) \\ & n3 : & c2 := 0 \\ & n4 : & wait(c1 = 1) \\ & n5 : & r2(*in\ crit\ sect*) \\ & n6 : & c2 := 1 \\ & n7 : & od \end{array}$$

Mutual Exclusion PTS

$V = \{pc1, pc2, c1, c2\}$, $F = \{m1, \ldots, m6, n1, \ldots, n6, 0, 1\}$

$$\begin{array}{rrcl} T = & pc1 = m1 & \rightarrow & pc1 := m2 \\ & pc1 = m2 & \rightarrow & pc1 := m3 \\ & pc1 = m3 & \rightarrow & (pc1, c1) := (m4, 0) \\ & pc1 = m4 \wedge c2 = 1 & to & pc1 := m5 \\ & pc1 = m5 & \rightarrow & pc1 := m6 \\ & pc1 = m6 & \rightarrow & (pc1, c1) := (m1, 1) \\ & pc2 = n1 & \rightarrow & pc2 := n2 \\ & pc2 = n2 & \rightarrow & pc2 := n3 \\ & pc2 = n3 & \rightarrow & (pc2, c2) := (n4, 0) \\ & pc2 = n4 \wedge c1 = 1 & to & pc2 := n5 \\ & pc2 = n5 & \rightarrow & pc2 := n6 \\ & pc2 = n6 & \rightarrow & (pc2, c2) := (n1, 1) \end{array}$$

$init = (pc1 = m1 \wedge pc2 = n1 \wedge c1 = 1 \wedge c2 = 1)$

Interpreting PTS as LTS

Let $(\mathcal{S}, T, init)$ be a program transition system. Assume V finite, $\mathcal{D}$ at most countable.

- Let $Q = V \times \mathcal{D}$, interpretted as all assingnments of values to variables
  - Can restrict to pairs (v, d) where v and d have same type
- Let $\sigma = T$
- Let $\delta(q, g \rightarrow (v_1, \ldots, v_n) := (e_1, \ldots, e_n), q') =$
  $\mathcal{M}_q(g) \wedge q'(v) = \begin{cases} \mathcal{T}_q(e_i) & \text{if } v = v_1 \text{ some } i \leq n \\ q(v) & \text{otherwise} \end{cases}$
- $I = \{q | \mathcal{T}_q(init) = \mathbf{T}\}$

Examples (cont)

- LTS for traffic light has $3 \times 3 \times 2 = 18$ possible well typed states
  - Is is possible to reach a state where $NSColor \neq Red \wedge EWColor \neq Red$ from an initial state?
  - If so, what sequence of actions alows this?
  - Do all the immediate predecessors of a state where $NSColor = Green \vee EWColor = Green$ satisfy $NSColor = Red \wedge EWColor = Red$?
  - If not, are any of those offend states reachable from and initial state, and if so, how?
- LTS for Mutual Exclusion has $6 \times 6 \times 2 \times 2 = 144$ posible well-tped states.
  - Is is possible to reach a state where $pc1 = m5 \wedge pc2 = n5$?
- How can we state these questions rigorously, formally?
- Cna we find an algorihm to answer these types of questions?

Linear Temporal Logic - Syntax

$$\begin{array}{rl} \varphi ::= & p | (\varphi) | \not\varphi | \varphi \wedge \varphi' | \varphi \vee \varphi' \\ | & \circ\varphi | \varphi \mathcal{U} \varphi' | \varphi \mathcal{V} \varphi' | \Box\varphi | \Diamond\varphi \end{array}$$

- p – a propostion over state variables
- $\circ\varphi$ – “next”
- $\varphi \mathcal{U} \varphi'$ – “until”
- $\varphi \mathcal{V} \varphi'$ – “releases”
- $\Box\varphi$ – “box”, “always”, “forever”
- $\Diamond\varphi$ – “diamond”, “eventually”, “sometime”

LTL Semantics: The Idea

Formal LTL Semantics

Given:

- $\mathcal{G} = (V, F, af, R, ar)$ signature expressing state propositions
- Q set of states,
- $\mathcal{M}$ modeling function over Q and cG: $cM(q, p)$ is true iff q models p. Write $q \models p$.
- $\sigma = q_0 q_1 \ldots q_n \ldots$ infinite sequence of state from Q.
- $sigma^i = q_i q_{i+1} \ldots q_n \ldots$ the i^{th} tail of σ

Say σ models LTL formula φ, write $\sigma \models \varphi$ as follows:

- $\sigma \models p$ iff $q_0 \models p$
- $\sigma \models \neg\varphi$ iff $\sigma \not\models \varphi$
- $\sigma \models \varphi \wedge \psi$ iff $\sigma \models \varphi$ and $\sigma \models \psi$.
- $\sigma \models \varphi \vee \psi$ iff $\sigma \models \varphi$ or $\sigma \models \psi$.
- $\sigma \models \circ\varphi$ iff $\sigma^1 \models \varphi$
- $\sigma \models \varphi \mathcal{U} \psi$ iff for some k, $\sigma^k \models \psi$ and for all $i \leq k$, $\sigma^i \models \varphi$
- $\sigma \models \varphi \mathcal{V} \psi$ iff for some k, $\sigma^k \models \varphi$ and for all $i < k$, $\sigma^i \models \psi$, or for all i, $\sigma^i \models \psi$.